

Appl. No. 09/699,517  
Declaration under 37 C.F.R. § 1.131  
Reply to Final Office Action of September 23, 2003



#14  
3-16-04

Appl. No. : 09/699,517  
Applicant : Timothy A. McDonough et al.  
Filed : October 31, 2000  
Title : User Notification System with an Illuminated Computer Input Device

TC/A.U. : 2674  
Examiner : Abbas I. Abdulsalam

Docket No. : 003797.00007

RECEIVED

MAR 15 2004

Technology Center 2600

Commissioner for Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

**DECLARATION UNDER 37 C.F.R. § 1.131**

Sir:

We, TIMOTHY A. McDONOUGH, CARL J. LEDBETTER, ROBERT SCOTT PLANK, STEVEN W. FISHER, STEVEN T. KANEKO, and STEVEN BATHICHE, hereby declare<sup>1</sup> that:

- 1) We are named as joint inventors of the above-captioned application, U.S. Application Serial No. 09/699,517, and all claims presently pending therein;
- 2) I, TIMOTHY A. McDONOUGH, am presently employed by Microsoft Corporation (Microsoft) and have been since July 1997. Microsoft is the assignee of the above-identified application.

<sup>1</sup> Each numbered declaration is a joint declaration unless an individual reference has been made. In such a case, the referenced individual is making the numbered declaration.

- 3) I, CARL J. LEDBETTER, am presently employed by Microsoft Corporation (Microsoft) and have been since January 1995. Microsoft is the assignee of the above-identified application.
- 4) I, ROBERT SCOTT PLANK, am presently employed by Microsoft Corporation (Microsoft) and have been since January 1992. Microsoft is the assignee of the above-identified application.
- 5) I, STEVEN W. FISHER, am presently employed by Microsoft Corporation (Microsoft) and have been since September 1997. Microsoft is the assignee of the above-identified application.
- 6) I, STEVEN T. KANEKO, am presently employed by Microsoft Corporation (Microsoft) and have been since September 1991. Microsoft is the assignee of the above-identified application.
- 7) I, STEVEN BATHICHE, am presently employed by Microsoft Corporation (Microsoft) and have been since June 1999. Microsoft is the assignee of the above-identified application.
- 8) We were employed by Microsoft during development of the above-identified invention.
- 9) We conceived of and reduced to practice the invention recited in at least a number of the claims of the above-captioned application prior to February 20, 2000.
- 10) Conception occurred prior to February 20, 2000, as is evidenced by the reproduction of a notebook entry in Exhibit A.
- 11) Actual reduction to practice occurred prior to February 20, 2000, as is evidenced by the source code in Exhibit B.
- 12) Support for at least claims 1-3, 6-10, 21, 27-28, and 30 of the above-captioned application can be found, among other places, at least within Exhibits A and B prepared prior to February 20, 2000.
- 13) The attached Exhibits A and B have not been materially altered since it was originally created except for the redaction of references to dates on the documents.

14) Each of us individually represents that we are over 18 years of age and of competent mind.

15) All statements made of our own knowledge are true and all statements made on information and belief are believed to be true; and further, these statements were made with the knowledge that willful, false statement so made are punishable by fine or imprisonment or both, under 18 U.S.C. § 1001 and that such willful, false statements may jeopardize the validity of the above-identified application or any patent issuing thereon.

Respectfully submitted,

\_\_\_\_\_  
Timothy A. McDonough,  
Microsoft Corporation

\_\_\_\_\_  
Date

\_\_\_\_\_  
Carl J. Ledbetter  
Microsoft Corporation

\_\_\_\_\_  
Date

\_\_\_\_\_  
Robert Scott Plank  
Microsoft Corporation

\_\_\_\_\_  
Date

  
\_\_\_\_\_  
Steven W. Fisher  
Microsoft Corporation

\_\_\_\_\_  
02/16/04  
Date

\_\_\_\_\_  
Steven T. Kaneko  
Microsoft Corporation

\_\_\_\_\_  
Date

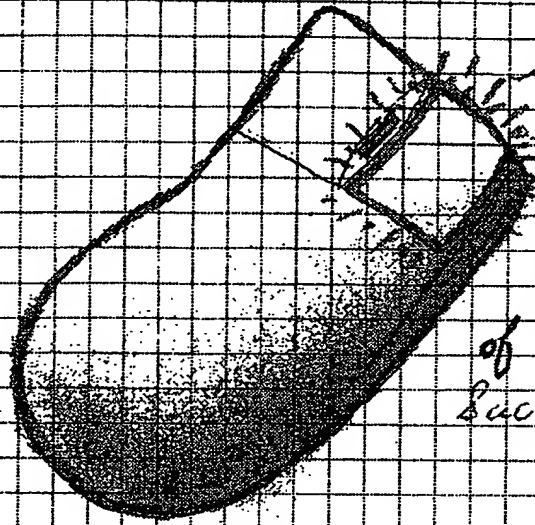
\_\_\_\_\_  
Steven Bathiche,  
Microsoft Corporation

\_\_\_\_\_  
Date

Appl. No. 09/699,517  
Declaration under 37 C.F.R. § 1.131  
Reply to Final Office Action of September 23, 2003

## **EXHIBIT A**

**Reproduction of Notebook Entry**



Light  
+  
Tactile  
Feedback  
for Notification  
of Email and  
Such

for any situation where the user needs  
to be made aware of a function  
for an object the button can  
light up + give a tactile  
response to tell the user they  
use me! or you have a new alert.

SUBJECT: Context menu inconsistency

WORK AND RECORD OF: Steve

WITNESSED AND UNDERSTOOD BY: [Signature]

WITNESSED AND UNDERSTOOD BY: [Signature]

PROJECT:

DATE:

DATE:

DATE:

Appl. No. 09/699,517  
Declaration under 37 C.F.R. § 1.131  
Reply to Final Office Action of September 23, 2003

## **EXHIBIT B**

**Source Code**

VERSION 5.00

Begin VB.Form form1

Caption = "HRD New Mail Notification"  
ClientHeight = 465  
ClientLeft = 60  
ClientTop = 345  
ClientWidth = 3750  
LinkTopic = "form1"  
ScaleHeight = 465  
ScaleWidth = 3750  
Visible = 0 'False

Begin VB.Timer Timer2

Interval = 100  
Left = 0  
Top = 0

End

Begin VB.Timer Timer1

Enabled = 0 'False  
Interval = 200  
Left = 480  
Top = 0

End

End

Attribute VB\_Name = "form1"

Attribute VB\_GlobalNameSpace = False

Attribute VB\_Creatable = False

Attribute VB\_PredeclaredId = True

Attribute VB\_Exposed = False

Option Explicit

Private Const KEYEVENTF\_EXTENDEDKEY = &H1

Private Const KEYEVENTF\_KEYUP = &H2

Private Const VK\_SCROLL = &H91

Private Const VK\_CAPITAL = &H14

Private Const VK\_NUMLOCK = &H90

Private messages As Integer

Private namespace As namespace

Private inbox As MAPIFolder

Private keystate() As Byte

Private result As Long

Private light\_state As Boolean

Private WithEvents m\_application As outlook.Application

Attribute m\_application.VB\_VarHelpID = -1

Private hwndMouseDriver As Long

Private light\_on As Long

Private light\_off As Long

Private Declare Function RegisterWindowMessage Lib "user32" Alias

"RegisterWindowMessageA" (ByVal lpString As String) As Long

Private Declare Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal  
hwnd As Long, ByVal wParam As Long, ByVal lParam As Any) As Long

```
Private Declare Function FindWindow Lib "user32" Alias "FindWindowA" (ByVal lpClassName As String, ByVal lpWindowName As String) As Long
Private Declare Function PostMessage Lib "user32" Alias "PostMessageA" (ByVal hwnd As Long, ByVal wParam As Long, ByVal lParam As Long) As Long
```

```
Private Declare Sub keybd_event Lib "user32" (ByVal bVk As Byte, ByVal bScan As Byte, ByVal dwFlags As Long, ByVal dwExtraInfo As Long)
Private Declare Function GetKeyboardState Lib "user32" (pbKeyState As Byte) As Long
```

```
Private Declare Function GetActiveWindow Lib "user32" () As Long
Private Declare Function GetWindowText Lib "user32" Alias "GetWindowTextA" _
    (ByVal hwnd As Long, ByVal lpString As String, ByVal cch As Long) As Long
```

```
Private Declare Function GetFocus Lib "user32" () As Long
```

```
'tray icon stuff
```

```
'Declare a user-defined variable to pass to the Shell_NotifyIcon function.
```

```
Private Type NOTIFYICONDATA
    cbSize As Long
    hwnd As Long
    uId As Long
    uFlags As Long
    uCallbackMessage As Long
    hIcon As Long
    szTip As String * 64
End Type
```

```
'Declare the constants for the API function. These constants can be found in the header file Shellapi.h.
```

```
'The following constants are the messages sent to the Shell_NotifyIcon function to add, modify, or delete an icon from the taskbar status area.
```

```
Private Const NIM_ADD = &H0
Private Const NIM_MODIFY = &H1
Private Const NIM_DELETE = &H2
```

```
'The following constant is the message sent when a mouse event occurs within the rectangular boundaries of the icon in the taskbar status area.
```

```
Private Const WM_MOUSEMOVE = &H200
```

```
'The following constants are the flags that indicate the valid members of the NOTIFYICONDATA data type.
```

```
Private Const NIF_MESSAGE = &H1
Private Const NIF_ICON = &H2
Private Const NIF_TIP = &H4
```

```
'The following constants are used to determine the mouse input on the icon in the taskbar status area.
```

```
'Left-click constants.
```



```

Private Const WM_LBUTTONDBLCLK = &H203 'Double-click
Private Const WM_LBUTTONDOWN = &H201 'Button down
Private Const WM_LBUTTONUP = &H202 'Button up

'Right-click constants.
Private Const WM_RBUTTONDBLCLK = &H206 'Double-click
Private Const WM_RBUTTONDOWN = &H204 'Button down
Private Const WM_RBUTTONUP = &H205 'Button up

'Declare the API function call.
Private Declare Function Shell_NotifyIcon Lib "shell32" _
    Alias "Shell_NotifyIconA" _
    (ByVal dwMessage As Long, pnid As NOTIFYICONDATA) As Boolean

'Dimension a variable as the user-defined data type.
Dim nid As NOTIFYICONDATA

```

```

'Private Declare Function RegisterWindowMessage Lib "user32" Alias
'RegisterWindowMessageA" (ByVal lpString As String) As Long
'Private Declare Function PostMessage Lib "user32" Alias "PostMessageA" (ByVal
hwnd As Long, ByVal wParam As Long, ByVal lParam As Long) As
Long

```

```

Private Sub Command1_Click()

```

```

End Sub

```

```

Private Sub scroll_light(state As Boolean)

```

```

    ReDim keystate(0 To 255)
    result = GetKeyboardState(keystate(0))
    If (result) Then
        If (keystate(VK_SCROLL) And Not (state)) Then
            keybd_event VK_SCROLL, 0, KEYEVENTF_EXTENDEDKEY Or 0, 0
            keybd_event VK_SCROLL, 0, KEYEVENTF_EXTENDEDKEY Or KEYEVENTF_KEYUP,
0
        ElseIf (Not (keystate(VK_SCROLL)) And state) Then
            keybd_event VK_SCROLL, 0, KEYEVENTF_EXTENDEDKEY Or 0, 0
            keybd_event VK_SCROLL, 0, KEYEVENTF_EXTENDEDKEY Or KEYEVENTF_KEYUP,
0
        End If
    End If

```

```

End Sub

```

```

Private Sub Form_Initialize()

light_on = RegisterWindowMessage("light_on")
light_off = RegisterWindowMessage("light_off")

hwndMouseDriver = FindWindow("POINTEXE", "Pointer.exe Invisible Window")

result = PostMessage(hwndMouseDriver, light_on, 0, 0)

Set m_application = CreateObject("outlook.application", "")
Set namespace = m_application.GetNamespace("MAPI")
Set inbox = namespace.GetDefaultFolder(olFolderInbox)

'if light on turn it off
'scroll_light (False)

If inbox.UnReadItemCount <> 0 Then
messages = inbox.UnReadItemCount * 2
Timer1.Enabled = True
End If

End Sub

Private Sub Form_KeyPress(KeyAscii As Integer)
scroll_light (False)
Timer1.Enabled = False
Timer2.Enabled = False

End Sub

Private Sub Form_Load()
'tray icon stuff

'Set the individual values of the NOTIFYICONDATA data type.
nid.cbSize = Len(nid)
nid.hwnd = form1.hwnd
nid.uId = vbNull
nid.uFlags = NIF_ICON Or NIF_TIP Or NIF_MESSAGE
nid.uCallbackMessage = WM_MOUSEMOVE
nid.hIcon = form1.Icon
nid.szTip = "Double Click to close Mouse Light" & vbNullChar

'Call the Shell_NotifyIcon function to add the icon to the taskbar
'status area.
Shell_NotifyIcon NIM_ADD, nid

'tray icon end

End Sub

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y
As Single)

Dim temp As Integer

```

```

    ' Return value that Windows has written to string.
    ActiveWindowCaption = strCaption
End If
End Function '

Private Sub Timer1_Timer()

If messages <> 0 And (messages Mod 2 = 0) Then
    result = PostMessage(hwndMouseDriver, light_on, 0, 0)
    'scroll_light (True)
    messages = messages - 1
ElseIf messages <> 0 And Not (messages Mod 2 = 0) Then
    result = PostMessage(hwndMouseDriver, light_off, 0, 0)
    ' scroll_light (False)
    messages = messages - 1
ElseIf inbox.UnReadItemCount <> 0 Then
    result = PostMessage(hwndMouseDriver, light_off, 0, 0)
    ' scroll_light (False)
    Timer1.Enabled = False
    Timer2.Enabled = True
Else
    result = PostMessage(hwndMouseDriver, light_on, 0, 0)
    ' scroll_light (False)
    Timer2.Enabled = False
    Timer1.Enabled = False
End If

End Sub

```

```

Private Sub Timer2_Timer()
    messages = inbox.UnReadItemCount * 2
    Timer2.Enabled = False
    Timer1.Enabled = True
End Sub

```

```

        'Event occurs when the mouse pointer is within the rectangular
        'boundaries of the icon in the taskbar status area.
        Dim msg As Long
        Dim sFilter As String
        msg = X / Screen.TwipsPerPixelX
        Select Case msg
            Case WM_LBUTTONDOWN
            Case WM_LBUTTONUP
            Case WM_LBUTTONDBLCLK
                Unload form1
            Case WM_RBUTTONDOWN
            Case WM_RBUTTONUP
            Case WM_RBUTTONDBLCLK
        End Select
    End Sub

Private Sub Form_Terminate()
    'Delete the added icon from the taskbar status area when the
    'program ends.
    Shell_NotifyIcon NIM_DELETE, nid
    result = PostMessage(hwndMouseDriver, light_on, 0, 0)
End Sub

Private Sub Form_Unload(Cancel As Integer)
    'Delete the added icon from the taskbar status area when the
    'program ends.
    Shell_NotifyIcon NIM_DELETE, nid
    result = PostMessage(hwndMouseDriver, light_on, 0, 0)

End Sub

Private Sub m_application_NewMail()

Timer1.Enabled = False
messages = inbox.UnReadItemCount * 2
result = PostMessage(hwndMouseDriver, light_off, 0, 0)
'scroll_light (False)
Timer1.Enabled = True

End Sub

Function ActiveWindowCaption() As String

    Dim strCaption As String
    Dim lngLen As Long

    ' Create string filled with null characters.
    strCaption = String$(255, vbNullChar)
    ' Return length of string.
    lngLen = Len(strCaption)

    ' Call GetActiveWindow to return handle to active window,
    ' and pass handle to GetWindowText, along with string and its length.
    If (GetWindowText(GetFocus, strCaption, lngLen) > 0) Then

```